CLAIMS

1. A graphics driver, the graphics driver receiving high level graphics data from a plurality of graphics clients, the graphics driver comprising:

   a partitioning controller, the partitioning controller allocating a periodic budget to each of the plurality of graphics clients;

   a graphics translator, the graphics translator receiving the high level graphics data from the plurality of graphics clients and converting the high level graphics data to a plurality of graphics primitives; and

   a chain builder, the chain builder creating a plurality of graphics primitive chains, each of the graphics primitive chains corresponding to at least one graphics client, the chain builder passing the plurality of graphics primitive chains to a graphics processor, the graphics primitive chains configured such that the graphics processor renders according to the periodic budget of each of the plurality of graphics clients.

2. The graphics driver of claim 1 wherein the plurality of graphics primitive chains are configured to have a total rendering time for each chain that corresponds to the periodic budget for the corresponding graphics client.

3. The graphics driver of claim 1 wherein each of the plurality of graphics primitive chains includes a command to restore context to the graphics processor before processing graphics primitives in the graphics primitive chain.

4. The graphics driver of claim 1 wherein each of the periodic budgets comprises a reoccurring portion of a reoccurring period.

5. The graphics driver of claim 1 further comprising a chain time estimator, the chain time estimator estimating processing time for each graphics primitive in each graphics primitive chain to determine when each graphics primitive chain should be terminated to make likely that each graphics primitive chain will be processed within the periodic budget for the corresponding graphics client.

6. The graphics driver of claim 5 further comprising a feedback adjustment mechanism, the feedback adjustment mechanism determining actual processing time for the graphics primitive chain and comparing to the estimated processing time to provide adjustment of chain time estimator.

7. The graphics driver of claim 6 wherein the feedback adjustment mechanism determines the actual processing time with a chain timer coupled to the feedback adjustment mechanism.

8. The graphics driver of claim 7 wherein each graphics primitive chain includes start packet to start the chain timer and a stop packet to stop the chain timer.

9. The graphics driver of claim 1 wherein the graphics driver enforces the periodic budget for each of the plurality of graphics clients.

10. The graphics driver of claim 9 wherein the graphics driver enforces the periodic budget for each of the plurality of graphics clients by measuring the processing time of each graphics primitive chain and resetting the graphics processor when the processing time for a graphics primitive chain will exceed its corresponding periodic budget.

11. The graphics driver of claim 10 wherein the graphics driver discards any remaining graphics primitive chains in a same scene as the graphics primitive chain that will exceed its corresponding periodic budget.

12. The graphics driver of claim 1 wherein each of the graphics primitive chains includes a jump packet, the jump packet pointing to itself when the graphics primitive chain is built and modified to point to a next graphics primitive chain when the next graphics primitive chain is built such that the next graphics primitive chain will be automatically dispatched when ready.

13. The graphics driver of claim 1 wherein the graphics driver loads a default context for a first chain of a new scene and maintains a context register set during rendering of the graphics primitive chains, such that when a scene requires multiple graphics primitive chains a context from a previous chain can be restored at the beginning of a subsequent chain.

14. A method for allocating processing time in a graphics processor, the method comprising the steps of:

a) allocating a periodic budget to each of a plurality of a graphics clients;

a) receiving high-level graphics data from the plurality of graphics clients;

b) translating the high-level graphics data into a plurality of graphics primitives;

c) building a plurality of graphics primitive chains, each of the graphics primitive chains corresponding to at least one of the plurality of graphics clients, each of the graphics primitive chains configured to allocate processing time according to the periodic budget of each of the plurality of graphics clients;

d) transferring the plurality of graphics primitive chains to the graphics processor; and

e) rendering the plurality of graphics primitive chains.

15. The method of claim 14 wherein each of the plurality of graphics primitive chains are configured to have a total rendering time that corresponds to the periodic budget for the corresponding graphics client.

16. The method of claim 14 further comprising the step of restoring context prior to rendering each of the plurality of graphics chains.

17. The method of claim 16 wherein the step of restoring context prior to rendering each of the plurality of graphics chains comprises loading a default context for a first primitive chain in a new scene.

18. The method of claim 16 wherein the step of restoring context prior to rendering each of the plurality of graphics chains comprises loading a maintained context from rendering of a previous chain.

19. The method of claim 14 wherein each of the periodic budgets comprises a reoccurring portion of a reoccurring period.

20. The method of claim 14 further comprising the step of estimating the processing time for each graphics primitive in each graphics primitive chain to determine when each graphics chain should be terminated to make likely that each graphics primitive chain will be processed with the periodic budget for the corresponding graphics client.

21. The method of claim 20 further comprising the step determining actual processing time for the graphics primitive chain and comparing to the estimated processing time to provide adjustment of the step of estimating the processing time.

22. The method of claim 14 further comprising the step of enforcing the periodic budget for each of the graphics clients.

23. The method of claim 22 wherein the step of enforcing the periodic budget for each of the graphics clients comprises measuring the processing time of each graphics primitive chain and resetting a graphics processor when the processing time for a graphics primitive chain will exceeds its corresponding budget.

24. The method of claim 23 further comprising the step of discarding any remaining graphics primitive chains in a same scene as a current graphics chain after resetting the graphics processor.

25. The method of claim 14 wherein each of the graphics primitive chains includes a jump packet, the jump packet pointing to itself when the graphics primitive chain is modified to point to a next graphics primitive chain when the next graphics primitive chain such that the next graphics primitive chain will be automatically dispatched when ready.

26. The method of claim 14 further comprising the step of loading a default context for a first chain of a new scene and maintaining a context register set during rendering of the graphics primitive chains, such that when a scene requires multiple graphics primitive chains a context from a previous chain can be restored at the beginning of a subsequent chain.

27. A program product comprising:

a) a graphics driver program, the graphics driver program receiving high level graphics data from a plurality of graphics clients, the graphics driver program including:

a partitioning controller, the partitioning controller allocating a periodic budget to each of the plurality of graphics clients;

a graphics translator, the graphics translator receiving the high level graphics data from the plurality of graphics clients and converting the high level graphics data to a plurality of graphics primitives; and

a chain builder, the chain builder creating a plurality of graphics primitive chains, each of the graphics primitive chains corresponding to at least one graphics client, the chain builder passing the plurality of graphics primitive chains to a graphics processor, the graphics primitive chains configured such that the graphics processor renders according to the periodic budget of each of the plurality of graphics clients; and

b) signal bearing media bearing the graphics driver program.

28. The program product of claim 27 wherein the plurality of graphics primitive chains are configured to have a total rendering time for each chain that corresponds to the periodic budget for the corresponding graphics client.

29. The program product of claim 27 wherein each of the plurality of graphics primitive chains includes a command to restore context to the graphics processor before processing graphics primitives in the graphics primitive chain.

30. The program product of claim 27 wherein each of the periodic budgets comprises a reoccurring portion of a reoccurring period.

31. The program product of claim 27 further comprising a chain time estimator, the chain time estimator estimating processing time for each graphics primitive in each graphics primitive chain to determine when each graphics primitive chain should be terminated to make likely that each graphics primitive chain will be processed within the periodic budget for the corresponding graphics client.

32. The program product of claim 31 further comprising a feedback adjustment mechanism, the feedback adjustment mechanism determining actual processing time for the graphics primitive chain and comparing to the estimated processing time to provide adjustment of chain time estimator.

33. The program product of claim 32 wherein the feedback adjustment mechanism determines the actual processing time with a chain timer coupled to the feedback adjustment mechanism.

34. The program product of claim 33 wherein each graphics primitive chain includes start packet to start the chain timer and a stop packet to stop the chain timer.

35. The program product of claim 27 wherein the graphics driver enforces the periodic budget for each of the plurality of graphics clients.

36. The program product of claim 35 wherein the graphics driver enforces the periodic budget for each of the plurality of graphics clients by measuring the processing time of each graphics primitive chain and resetting the graphics processor when the processing time for a graphics primitive chain will exceed its corresponding periodic budget.

37. The program product of claim 36 wherein the graphics driver discards any remaining graphics primitive chains in a same scene as the graphics primitive chain that will exceed its corresponding periodic budget.

38. The program product of claim 27 wherein each of the graphics primitive chains includes a jump packet, the jump packet pointing to itself when the graphics primitive chain is built and modified to point to a next graphics primitive chain when the next graphics primitive chain is built such that the next graphics primitive chain will be automatically dispatched when ready.

39. The program product of claim 27 wherein the graphics driver loads a default context for a first chain of a new scene and maintains a context register set during rendering of the graphics primitive chains, such that when a scene requires multiple graphics primitive chains a context from a previous chain can be restored at the beginning of a subsequent chain.

40. The program product of claim 27 wherein the signal bearing media comprises recordable media.

41. The program product of claim 27 wherein the signal bearing media comprises transmission media.